AMENDMENTS TO THE CLAIMS

This listing of claims replaces all prior versions, and listings, of claims in the application:

1. (Original) A method comprising:

storing first tuples in a first table in a database system;

storing second tuples in a second table in the database system;

partitioning the first and second tuples into plural portions;

redistributing the first and second tuples to plural nodes according to the partitioning; and

hash joining the first and second tuples to produce result tuples as the first and second tuples are being redistributed to the plural nodes.

2. (Cancelled)

3. (Original) The method of claim 1, further comprising:

retrieving the result tuples at random.

4. (Original) The method of claim 1, hash joining the first and second tuples to produce result tuples as the first and second tuples are being redistributed to the plural nodes further comprising:

producing result tuples at one of the plural nodes; and

simultaneously producing result tuples at a second of the plural nodes.

5. (Original) The method of claim 1, wherein redistributing the first and second tuples to plural nodes comprises redistributing based on split vectors containing predefined ranges.

6. (Original) The method of claim 5, wherein partitioning the first and second tuples into plural portions comprises:

partitioning first and second tuples into hash tables in each node.

7. (Original) The method of claim 6, wherein hash joining the first and second tuples comprises:

allocating a portion of a memory to a first hash table;

allocating a second portion of the memory to a second hash table; and

hash joining first tuples in the first hash table with second tuples in the second hash table.

8. (Original) The method of claim 7, wherein hash joining the first and second tuples comprises:

determining that the portion of the memory allocated to the first hash table is full;

allocating a stable storage to the first hash table; and

storing first tuples in the stable storage.

9. (Original) The method of claim 8, further comprising:

continuing to store second tuples in the second hash table; and

hash joining second tuples in the second hash table with first tuples in the first hash table.

10. (Original) The method of claim 9, further comprising:

determining that the second portion of the memory allocated to the second hash table is full;

allocating a second stable storage to the second hash table;

storing second tuples in the second stable storage; and

hash joining second tuples in the second stable storage with first tuples in the first hash table.

11. (Original) The method of claim 10, wherein hash joining the first and second tuples comprises:

generating a third hash table once all first tuples and second tuples are redistributed to each node;

retrieving one of the first tuples from the stable storage;

hash joining the one of the first tuples with tuples in the second hash table; and

storing the one of the first tuples in the third hash table.

12. (Original) The method of claim 11, further comprising:

retrieving one of the second tuples from the second stable storage; and

hash joining the one of the second tuples with tuples in the third hash table.

13. (Previously Presented) A database system comprising:

a plurality of nodes; and

instructions for enabling the database system to:

store first tuples in a first table distributed across the plurality of nodes;

store second tuples in a second table distributed across the plurality of nodes;

partition the first and second tuples into plural portions;

redistribute the first and second tuples to the plurality of nodes according to the partitioning; and

hash join the first and second tuples to produce result tuples as the first and second tuples are being redistributed to the plurality of nodes.

14. (Cancelled)

15. (Previously Presented) The database system of claim 13, wherein the result tuples are available at random.

16. (Previously Presented) The database system of claim 13, wherein each node comprises a memory, and wherein the instructions further partition the first and second tuples into plural portions by:

partitioning first tuples into first hash tables; and

partitioning second tuples into second hash tables, wherein the hash tables are in the memory.

17. (Previously Presented) The database system of claim 16, wherein the instructions further:

allocate a portion of the memory to the first hash table;

allocate a second portion of the memory to the second hash table; and

hash join first tuples in the first hash table with second tuples in the second hash table.

18. (Previously Presented) The database system of claim 17, wherein the instructions further:

determine that the portion of the memory allocated to the first hash table is full; and

store first tuples in a stable storage.

19. (Previously Presented) The database system of claim 18, wherein the instructions further:

continue to store second tuples in the second hash table; and

hash join second tuples in the second hash table with first tuples in the first hash table.

20. (Previously Presented) The database system of claim 19, wherein the instructions further:

determine that the second portion of the memory allocated to the second hash table is full;

allocate a second stable storage to the second hash table;

store second tuples in the second stable storage; and

hash join second tuples in the second stable storage with first tuples in the first hash table.

21. (Previously Presented) The database system of claim 20, wherein the instructions further:

generate a third hash table once all first tuples and second tuples are redistributed to each node;

retrieve one of the first tuples from the stable storage;

hash join the one of the first tuples with tuples in the second hash table; and

store the one of the first tuples in the third hash table.

22. (Previously Presented) The database system of claim 21, wherein the instructions further:

retrieve one of the second tuples from the second stable storage; and

hash join the one of the second tuples with tuples in the third hash table.

23. (Previously Presented) An article comprising a medium storing instructions for enabling a processor-based system to:

store first tuples in a first table in a database system;

store second tuples in a second table in the database system;

partition the first and second tuples into plural portions;

redistribute the first and second tuples to plural nodes of the database system according to the partitioning; and

hash join the first and second tuples to produce result tuples as the first and second tuples are being redistributed to the plural nodes.

24. (Original) The article of claim 23, further storing instructions for enabling a processor-based system to:

retrieving the result tuples once the hash join is performed.

25. (Original) The article of claim 24, further storing instructions for enabling a processor-based system to:

redistribute based on split vectors containing predefined ranges.

26. (Original) The article of claim 25, further storing instructions for enabling a processor-based system to:

partition first and second tuples into hash tables in each node.

27. (Original) The article of claim 26, further storing instructions for enabling a processor-based system to:

allocate a portion of a memory to a first hash table;

allocate a second portion of the memory to a second hash table; and

hash join first tuples in the first hash table with second tuples in the second hash table.

28. (Original) The article of claim 27, further storing instructions for enabling a processor-based system to:

determine that the portion of the memory allocated to the first hash table is full; and

store first tuples in a stable storage.

29. (Original) The article of claim 28, further storing instructions for enabling a processor-based system to:

continue to store second tuples in the second hash table; and

hash join second tuples in the second hash table with first tuples in the first hash table.

30. (Original) The article of claim 29, further storing instructions for enabling a processor-based system to:

determine that the second portion of the memory allocated to the second hash table is full;

allocate a second stable storage to the second hash table;

store second tuples in the second stable storage; and

hash join second tuples in the second stable storage with first tuples in the first hash table.

31. (Original) The article of claim 30, further storing instructions for enabling a processor-based system to:

generate a third hash table once all first tuples and second tuples are redistributed to each node;

retrieve one of the first tuples from the stable storage;

hash join the one of the first tuples with tuples in the second hash table; and

store the one of the first tuples in the third hash table.

32. (Original) The article of claim 31, further storing instructions for enabling a processor-based system to:

retrieve one of the second tuples from the second stable storage; and

hash join the one of the second tuples with tuples in the third hash table.

33. (Previously Presented) The method of claim 1, wherein storing the first tuples in the first table comprises distributing the first tuples across the plural nodes of the database system, and wherein storing the second tuples in the second table comprises distributing the tuples across the plural nodes.

34. (Previously Presented) The method of claim 33, wherein redistributing the first and second tuples comprises redistributing the first and second tuples to the plural nodes of the database system.

35. (Previously Presented) The article of claim 23, wherein storing the first tuples in the first table comprises storing the first tuples in the first table distributed across the plural nodes of the database system, and wherein storing the second tuples in the second table comprises storing the second tuples distributed across the plural nodes of the database system.

36. (New) The method of claim 1, wherein each of the nodes contains a first hash table to receive first tuples, and a second hash table to receive second tuples, the method further comprising:

storing redistributed first tuples in respective first hash tables; and

storing redistributed second tuples in respective second hash tables.

37. (New) The method of claim 36, wherein hash joining first tuples and second tuples comprises hash joining first tuples and second tuples from corresponding first and second hash tables.

38. (New) The database system of claim 13, wherein each of the nodes contains a first hash table to receive first tuples, and a second hash table to receive second tuples,

wherein the instructions further:

store redistributed first tuples in respective first hash tables; and

store redistributed second tuples in respective second hash tables.

39. (New) The database system of claim 38, wherein the instructions further hash join the first tuples and the second tuples from corresponding first and second hash tables.

40. (New) The article of claim 23, wherein each of the nodes contain a first hash table to receive first tuples, and a second hash table to receive second tuples, wherein the instructions when executed cause the processor-based system to further:

store redistributed first tuples in respective first hash tables; and

store redistributed second tuples in respective second hash tables.

41. (New) The article of claim 40, wherein hash joining first tuples and second tuples comprises hash joining first tuples and second tuples from corresponding first and second hash tables.